(12) **PATENT**
(19) **AUSTRALIAN PATENT OFFICE**

(11) Application No. **AU 200071842 B2**
(10) Patent No. **744936**

(54) Title
Spread decision error diffusion

(51)[7] International Patent Classification(s)
H04N 001/405 **G06T 005/00**

(21) Application No: **200071842**

(22) Application Date: **2000.11.27**

(30) Priority Data

(31) Number	(32) Date	(33) Country
PQ4914	**1999.12.24**	**AU**

(43) Publication Date : **2001.06.28**
(43) Publication Journal Date : **2001.06.28**
(44) Accepted Journal Date : **2002.03.07**

(71) Applicant(s)
Canon Kabushiki Kaisha

(72) Inventor(s)
Peter Mitchell William Ilbery

(74) Agent/Attorney
SPRUSON and FERGUSON,GPO Box 3898,SYDNEY NSW 2001

Abstract

SPREAD DECISION DIFFUSION ERROR DIFFUSION

A method of halftoning an image is disclosed, said method comprising, for a current pixel, the steps of determining a modified neighbourhood error value (954) using a neighbourhood error value (917) and, when current pixel input value is substantially within a range of values, a neighbourhood error value (966, 996) of neighbouring pixels. An output value (908) is determined using a sum of the current pixel input value (900) and the modified neighbourhood error value (954), thereby determining an error value (920) as the difference between, firstly, the sum of the current pixel input value (913) and the current pixel neighbourhood error value (915), and secondly the output value of the current pixel (910). The error value for the current pixel is added to the neighbourhood error values of yet to be processed neighbouring pixels.

S&F Ref: 524262

AUSTRALIA

PATENTS ACT 1990

COMPLETE SPECIFICATION

FOR A STANDARD PATENT

ORIGINAL

Name and Address of Applicant :	Canon Kabushiki Kaisha 30-2, Shimomaruko 3-chome, Ohta-ku Tokyo 146 Japan
Actual Inventor(s):	Peter Mitchell William Ilbery
Address for Service:	Spruson & Ferguson St Martins Tower 31 Market Street Sydney NSW 2000
Invention Title:	Spread Decision Error Diffusion

ASSOCIATED PROVISIONAL APPLICATION DETAILS

[33] Country	[31] Applic. No(s)	[32] Application Date
AU	PQ4914	24 Dec 1999

The following statement is a full description of this invention, including the best method of performing it known to me/us:-

5815c

SPREAD DECISION ERROR DIFFUSION

Technical Field of the Invention

The present invention relates generally to the field of digital image processing, and in particular, to apparatus and method for digital halftoning continuous tone images. The invention also relates to a computer program product including a computer readable medium having recorded thereon a computer program for digital halftoning continuous tone images.

Background Art

An image is typically represented digitally as a rectangular array of pixels with each pixel having one of a restricted set of legitimate pixel values. Digital images may be black and white in which case the restricted set of legitimate pixel values is used to encode an optical density or luminance score. Digital images may also be colour, in which case the restricted set of pixel values typically encode an optical density or intensity score for each of a number of colour channels, for example Cyan, Magenta, Yellow and Black (CMYK) or Red, Green and Blue (RGB).

Digital images are often utilised in cases where the image value per pixel per colour channel is an 8 bit unsigned number, thereby providing intensity values in the range 0 through 255. Such images are often called "continuous tone" images because of the reasonably large number (256) of legitimate intensity values.

However, such digital images are often required to be printed on devices which provide a more limited variation in intensity or colour representation per pixel. For example, many Bubble Jet printers only provide the ability to print or not print a dot of each of a Cyan, Magenta, Yellow and Black ink at each pixel position.

In order to print a digital image on a printer which has a lower colour resolution than the digital image itself, it is necessary to use the original image to generate an image with the required lower colour resolution such that the generated image has a similar appearance to the original image. This process of generating a digital image of similar

appearance where each pixel colour value is within a smaller set of legitimate pixel colour values, is known as digital halftoning.

Digital halftoning can, in addition to use with the aforementioned monochrome bi-level case, be applied to instances where the halftoned image pixels have more than 2 legitimate output values (ie multi-level halftoning).

The case of a single colour channel, 8 bit per pixel, image which is halftoned to generate a 1 bit per pixel (bi-level) image which is printable on a black and white bubble jet printer is considered for ease of explanation. The input image values are known as greyscale values or grey levels. It is apparent however that this is illustrative and not restrictive. Each pixel of the halftoned image has one of 2 legitimate pixel values, designated a "no dot" value and a "dot" value.

An image region in the halftoned image will print as a minimum optical density ("fully white") region when all pixels of the region have the "no dot" value corresponding to non-placement of an ink dot. The image will print as a maximum optical density ("fully black") region when all pixels of the region have the "dot" value corresponding to placement of an ink dot. Finally, it will print as an intermediate optical density (halftone) region when some of the pixels of the region have the "no dot" value and some of the pixels have the "dot" value. A highlight (near white) region in the halftoned image will have only a few of the pixels with the "dot" value, and accordingly, in these regions the "dot" halftone output value is the "minority" or "exceptional" result. A shadow (near black) region will have only a few of the pixels with the "no dot" value, and accordingly, in these regions the "no dot" halftone output value is the minority or exceptional result.

The role of the halftoning process is to generate the printable image so that an appropriate number of ink dots will print in appropriate pixel positions so that there is a good match between the optical density of image regions in the original image and the average optical density of the matching image regions in the printed image.

Error diffusion is a well-known digital halftoning method originally developed by Floyd and Steinberg and described in the publication "An Adaptive Algorithm for Spatial Greyscale", Proceedings of the SID 17/2, pp 75-77 (1976). Error diffusion as

developed by Floyd and Steinberg is hereafter described as "standard error diffusion", or "SED".

SED processes pixels line by line from the top of the image to the bottom of the image. Each line (or "scanline") is processed one pixel at a time from left to right. The method employs a pixel decision rule in which a modified input image pixel value is compared against a threshold value. The pixel's modified input value is the sum of the pixel input image value and a neighbourhood error value for the pixel, and the neighbourhood error at a pixel is the sum of the errors distributed to that pixel from its previously processed neighbouring pixels. If the pixel's modified input value is less than the threshold, then the pixel's halftone output value is assigned to be the lower halftone output value, and if the modified input value is greater than the threshold, then the pixel's halftone output value is assigned to be the higher halftone output value.

Following determination of the pixel halftone output value, an error is determined for the pixel, this error being the difference between the pixel's modified input value and the pixel's halftone output value. The error is distributed to neighbouring, as yet unprocessed, pixels according to a set of weighting coefficients. The set of weighting coefficients is known as an error diffusion mask. The error at a pixel which is distributed to the pixel's neighbours can be considered as a sum of the neighbourhood error at the pixel, and the pixel-only error. The pixel-only error is the pixel's input value less the pixel's halftone output value.

Fig. 1 shows the error diffusion mask described by Floyd and Steinberg. The error at a current pixel 104 is distributed in the proportions 7/16, 1/16, 5/16 and 3/16 to the 4 neighbouring, as yet unprocessed, pixels 108, 110, 112 and 114 respectively. Pixels are processed in a direction depicted by an arrow 106. Previously processed pixels are shaded (as depicted by the reference numeral 102) in the region above the dark line 124. As yet unprocessed pixels whose neighborhood error will include an error contribution from the current pixel, either directly or indirectly, are shown shaded (as depicted by the reference numeral 116) in the region below the dark line 124. The sum of the weighting coefficients is 1, and as a result, 100% of the error of the current pixel 104 is transferred

to neighbouring pixels 108 to 114. If the sum of weighting coefficients were greater than 1, then the error would be amplified and could build up without bound. If on the other hand the sum of the weighting coefficients were less than 1, then the error would be reduced. By having the sum of weighting coefficients equal to unity, the average intensity of regions of the halftoned image tends to match the average intensity of corresponding regions in the input image, this being a very desirable characteristic of a halftoning process.

In implementations of standard error diffusion where multiplication of error by a weighting coefficient results in significant rounding errors, it is necessary to co-ordinate the calculation of error distributions to neighbouring pixels so that effectively 100% of current pixel error is transferred on. This can be achieved by determining error distributions from the current pixel 104 to all but one of the neighbouring pixels, say 108 – 112 by multiplication by a weight, and determining the error distribution to the remaining neighbouring pixel, say 114, by subtracting the other error distributions from the total error to be distributed. For improved execution speed, error distributions corresponding to a particular pixel error value are often determined in advance and retrieved from a look up table.

In the Floyd and Steinberg error diffusion mask, there are, in respect of the current pixel 104, 4 neighbouring pixels 108 – 114 which receive error distributions therefrom. The neighbouring pixels 108 – 114 are on a current scanline 120 and a succeeding scanline 122 only. An implementation of error diffusion with this mask requires the use of a single "line store" memory to store error sum values. The memory is referred to as a "line store" because it is typically required to store an error sum value for each pixel of a scanline. Prior art modifications to error diffusion which are described in this patent typically incur a cost, in that extra memory, which is associated with an additional one or more line stores, is required.

Standard error diffusion suffers from a disadvantage whereby in image regions of very low or very high intensity, SED generates a pattern of image values in the halftoned image which are poorly spread, exceptional values being concentrated in wavy

lines. It is recalled that a highlight (near white) region in the halftoned image will have only a few of the pixels with the "dot" value, and accordingly, in these regions the "dot" halftone output value is the minority or exceptional result. Similarly, a shadow (near black) region will have only a few of the pixels with the "no dot" value, and accordingly, in these regions the "no dot" halftone output value is the minority or exceptional result. These wavy line patterns which the exceptional values take on can be very noticeable and distracting to a viewer of the image, and are often known as "worm" artifacts.

Fig. 2 shows a section 200 of a halftoned image generated using standard error diffusion which shows worm artifact 202. The "worms" are the strings of black dots lining up in an approximately North East to South West direction.

Several modifications to error diffusion have been developed to reduce worm artifacts. One such method of reducing worm artifacts is by addition of some randomisation to the image. This can be achieved by, for example, adding noise to the input image, adding noise to the thresholds, and/or randomising the error distribution to neighbouring pixels. A large amount of noise or randomisation can be added to fully avoid worm artifacts, however, this also seriously degrades the halftoned image.

An alternate method of reducing worm artifacts is to vary the direction in which scanlines are processed. Yet another method of reducing worm artifacts is by use of larger error diffusion masks. Thus, for example, a large or a small error diffusion mask can be used depending on the input image grey level. The large error diffusion mask is suited to use in image regions of very high or very low greyscale, reducing the worm artifacts in those regions. A disadvantage of this method is that large error diffusion masks which distribute error to pixels of more than 1 succeeding scanline require additional error line stores and associated processing.

Complete prevention of worm artifacts can also be achieved. One published method of preventing worm artifacts is by modulating threshold values using a threshold imprint function. In this approach, the error diffusion threshold is adjusted as a function of both the halftone output and the input intensity by using the threshold impulse function. This method however requires use of memory for an additional line store to

store threshold adjustment values generated by preceding scanlines for use by a current scanline. The method also requires additional processing including addition of threshold impulse values and dampening of threshold values transferred to subsequent scanlines. An alternate published method for preventing worm artifacts uses threshold modulation based upon a serpentine error diffusion kernel. This method however also requires use of an additional line store memory to store threshold adjustment values. Additional processing is also required to diffuse threshold adjustment values. Yet another published method of preventing worm artifacts is the addition of periodic noise to the input image dependent upon filtered input image values. This however requires memory for additional line stores to store several neighbouring scanlines of processed input image data from which filtered input image values for a current scanline are determined. The filtered input image values are used in turn to determine the noise to be added to the input image data. A further published method of preventing worm artifacts is by imposing output position constraints. This however requires memory for additional lines stores to store halftoned image data for several previously processed scanlines. This modification also includes processing to exclude minority halftone output results when that result is present in a certain portion of the previously processed scanlines.

In many cases the desirability of a halftoning method is determined by how fast it executes and how easy it is to implement. For example, in software implementations in a printer driver on a general purpose computer, the execution speed is very important, whereas in special purpose hardware, the complexity and memory usage of the method are very important because they relate strongly to the expense of the circuitry.

The prior art modifications listed above which prevent worm artifacts in error diffusion all require use of additional line store memory together with the processing associated with use of that additional memory. Use of additional line store memory by a halftoning method generally indicates that it will execute slower in software and is more expensive to implement in special purpose hardware.

Disclosure of the Invention

It is an object of the present invention to overcome, or at least substantially ameliorate, one or more disadvantages of existing arrangements.

According to a first aspect of the invention, there is provided a method of halftoning an image, said method comprising, for a current pixel, when a current pixel input value is within a range of values, the steps of:

determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and a neighbourhood error value of one or more neighbouring pixels;

determining an output value using a sum of the current pixel input value and the modified neighbourhood error value;

determining an error value as the difference between, firstly, the sum of the current pixel input value and the current pixel neighbourhood error value, and secondly the output value of the current pixel; and

adding proportions of the error value for the current pixel to the neighbourhood error values of as yet unprocessed pixels.

According to another aspect of the invention, there is provided a method of halftoning an image, said image comprising a plurality of pixels each having an input value and an assignable output value that can take on one of at least two output values, wherein processing for each pixel comprises the steps of:

determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and, when the pixel input value is within a critical range of values, further using at least one neighbourhood error value of a neighbouring pixel;

determining the output value of a current pixel using a sum of the input value of the current pixel and the modified neighbourhood error value for the pixel;

determining an error value for the current pixel as the difference between, firstly, the sum of the input value of the current pixel and the neighbourhood error value for the pixel, and secondly the output value of the pixel; and

adding proportions of the error value for the current pixel to the neighbourhood error values of yet to be processed neighbouring pixels.

According to another aspect of the invention, there is provided an apparatus adapted for halftoning an image, said apparatus comprising:

first determining means for determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and a neighbourhood error value of one or more neighbouring pixels;

second determining means for determining an output value using a sum of the current pixel input value and the modified neighbourhood error value;

third determining means for determining an error value as the difference between, firstly, the sum of the current pixel input value and the current pixel neighbourhood error value, and secondly the output value of the current pixel; and

adding means for adding proportions of the error value for the current pixel to the neighbourhood error values of as yet unprocessed pixels.

According to another aspect of the invention there is provided an apparatus adapted for halftoning an image, said apparatus image comprising:

first determining means for determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and, when the pixel input value is within a critical range of values, further using at least one neighbourhood error value of a neighbouring pixel;

second determining means for determining the output value of a current pixel using a sum of the input value of the current pixel and the modified neighbourhood error value for the pixel;

third determining means for determining an error value for the current pixel as the difference between, firstly, the sum of the input value of the current pixel and the neighbourhood error value for the pixel, and secondly the output value of the pixel; and

adding means for adding proportions of the error value for the current pixel to the neighbourhood error values of yet to be processed neighbouring pixels.

According to another aspect of the invention there is provided a computer readable memory medium for storing a program for apparatus adapted for halftoning an image, said program comprising, for a current pixel, when a current pixel input value is within a range of values, the steps of:

code for a first determining step for determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and a neighbourhood error value of one or more neighbouring pixels;

code for a second determining step for determining an output value using a sum of the current pixel input value and the modified neighbourhood error value;

code for a third determining step for determining an error value as the difference between, firstly, the sum of the current pixel input value and the current pixel neighbourhood error value, and secondly the output value of the current pixel; and

code for an adding step for adding proportions of the error value for the current pixel to the neighbourhood error values of as yet unprocessed pixels.

According to another aspect of the invention there is provided a computer readable memory medium for storing a program for apparatus adapted for halftoning an image, said image comprising a plurality of pixels each having an input value and an assignable output value that can take on one of at least two output values, wherein said program comprises, for each pixel, the steps of:

code for a first determining step for determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and, when the pixel input value is within a critical range of values, further using at least one neighbourhood error value of a neighbouring pixel;

code for a second determining step for determining the output value of a current pixel using a sum of the input value of the current pixel and the modified neighbourhood error value for the pixel;

code for a third determining step for determining an error value for the current pixel as the difference between, firstly, the sum of the input value of the current pixel and

the neighbourhood error value for the pixel, and secondly the output value of the pixel; and

code for an adding step for adding proportions of the error value for the current pixel to the neighbourhood error values of yet to be processed neighbouring pixels.

Brief Description of the Drawings

A number of preferred embodiments of the present invention will now be described with reference to the drawings, in which:

Fig. 1 shows an error diffusion mask for standard error diffusion;

Fig. 2 depicts a halftone pattern with worm artifacts resulting from using the mask of Fig. 1 with standard error diffusion;

Fig. 3 depicts an influence function for error diffusion according to Fig. 1;

Fig. 4 shows error transfer to a pixel window according to an embodiment of the invention;

Fig. 5 depicts a halftone pattern produced in accordance with a preferred embodiment;

Fig. 6 illustrates determination of modified neighborhood error according to a preferred embodiment;

Fig. 7 depicts a prior art error diffusion process;

Fig. 8 illustrates neighborhood error distribution in relation to Fig. 7;

Fig. 9 depicts an error diffusion process in accordance with the preferred embodiment; and

Fig. 10 is a schematic block diagram of a general purpose computer upon which the preferred embodiment of the present invention can be practiced.

Detailed Description including Best Mode

Where reference is made in any one or more of the accompanying drawings to steps and/or features, which have the same reference numerals, those steps and/or features have for the purposes of this description the same function(s) or operation(s), unless the contrary intention appears.

A modification to error diffusion is described which removes worm artifacts in highlight and shadow regions. This is achieved with a very small additional amount of memory (but no additional line store memory) and only minor additional processing. Accordingly, the error diffusion processing is essentially unchanged for image regions which are not highlight or shadow regions.

With standard error diffusion, if the modified input value of a current pixel is lower than the threshold, then the halftone result for the current pixel is set low and the error distributed to neighbouring pixels acts to increase the likelihood that those neighbouring pixels are set high. Conversely if the modified input value of a current pixel is higher than the threshold, then the halftone result thereof is set high, and the error distributed to neighbouring pixels acts to increase the likelihood that those neighbouring pixels are set low. The proportion of the (non-zero) error of an arbitrary reference pixel which contributes to the neighbourhood error of a particular subsequently processed pixel of interest, is a measure of a disincentive to assign the pixel of interest the same halftone output result as the reference pixel. In other words, error distributed from the current pixel to a subsequent pixel acts to prevent that subsequent pixel being assigned the same halftone value as the current pixel, and the larger the distributed error, the less likely the subsequent pixel will be assigned that same halftone value.

A pixel contributes a proportion of its error to the neighbourhood error of pixels which are beyond those to which it directly distributes error. This is because of the sequential processing of pixels, and because the processing of each pixel includes the distribution of that pixel's error, being the sum of that pixel's neighbourhood error (the error distributed to that pixel) and that pixel's pixel-only error, according to the error diffusion mask. The proportion of error which a reference pixel contributes to the neighbourhood error of a subsequently processed pixel can be considered as an error impulse response or as an "influence". For many error diffusion methods, this influence is a function of the displacement of the subsequently processed pixel of interest from the reference pixel, the pixel processing order, and the error diffusion mask. For a particular processing order and error diffusion mask, an influence function can be defined which

associates an influence value with each pixel displacement from the reference pixel. For standard error diffusion, the influence function is defined over the set of integer pixel displacements (x, y) from a reference pixel by the following set of equations:

influence[x][y] = 0 for displacements (x, y) corresponding to pixels
which are processed prior to the reference pixel;
influence[x][y] = 1 for displacement (0, 0) corresponding to the reference pixel;
influence[x][y] = 0 for displacements (x, y) corresponding to pixels
which are processed after the reference pixel,
but with x being a very large negative number;
influence[x][y] = 7/16 * influence[x-1][y] + 1/16 * influence[x-1][y-1]
+ 5/16 * influence[x][y-1] + 3/16 * influence[x+1][y-1]
for other displacements;

where:

influence[x][y] is the influence of a current pixel on a pixel displaced [x][y] from the reference pixel.

x,y are the horizontal and vertical displacements of the pixel of interest to the right and down from the reference pixel;

Fig. 3 depicts an influence function 300 for standard error diffusion. The three dimensional function 300 is represented in terms of an influence magnitude [IM] axis 302, a scanline displacement [SD] axis 304 having units of scanlines from a current pixel, and a pixel displacement [PD] axis 306 having units of pixels along a scanline containing the current pixel. A peak 318 of the influence function 300 has a magnitude of unity (designated 320), at coordinate [SD][PD] = [0][0], and accordingly, the influence of the current pixel is unity (ie maximum) at the current pixel itself. Similarly, the influence of the current pixel at coordinate [SD][PD] = [8][4] (depicted by the reference numeral 308) lies between the values 0.1 and 0.2 as depicted by the legend reference numeral 316. In standard error diffusion, 100% of a pixel's error is distributed to neighbouring, as yet unprocessed, pixels on the current and succeeding scanlines. Furthermore, 100% of the

error which is distributed to a pixel (ie., the pixel's neighbourhood error) is, once the halftone result for the pixel has been determined, transferred on to neighbouring, as yet unprocessed pixels. This means that once a scanline has been processed, effectively 100% of the error of a pixel of that scanline has been transferred on to pixels of succeeding scanlines.

With standard error diffusion, the proportion of a pixel's error which is distributed to successive pixels along a scanline reduces exponentially. This means that the influence function reduces rapidly horizontally (depicted by an arrow 322), but only slowly vertically (depicted by an arrow 324).

Further, because of the unidirectional left to right scanline processing in standard error diffusion, there is left-right asymmetry in the influence function, as shown by the angular displacement between an arrow 326, and the [SD] axis direction as depicted by the arrow 324. One characteristic of this left-right asymmetry is that non-zero values of the influence function extend without bound to the right of the arrow 326, but do not extend to the left of the line inclined at 45 degrees to the scanline and extending South West of the reference pixel as depicted by an arrow 328. This can be seen more conveniently returning to Fig. 1, where the pixels influenced by a current pixel are shown shaded with dots 116 falling below a dark line 124. The unshaded region 118 depicts the area in which the influence function takes on a zero value, extending to the West and Southwest of the current pixel 104.

Very low intensity input image regions and very high intensity input image regions are both characterised by very few pixels having the minority or exceptional halftone output result. In other words, the dot patterns for these image regions are sparse and determination of halftone output values of pixels within such regions depends significantly on errors of a wide neighbourhood of previously processed pixels.

It is noted that worm artifacts occur mainly in input image regions with very high and very low image values.

It is reasonable to hypothesise that worm artifacts are due to the combination of:

1) non-ideal characteristics of the influence function which become more pronounced at large pixel displacements, including, firstly, dissimilar horizontal and vertical reduction, and secondly, aspects of left-right asymmetry;

2) the fact that the determination of halftone output values for very low intensity and very high intensity image regions depend significantly on influence function values at distant positions.

The disclosed modification to standard error diffusion counters the effects of the poor shape of the influence function at large pixel displacements by expanding the basic pixel decision rule of standard error diffusion. Accordingly, not only the error transferred to the current pixel is considered, but also the errors transferred to pixels in a horizontal window around the current pixel are considered.

Fig. 4 shows a diagram of errors distributed to pixels 416-418 within a horizontal window 420 around the current pixel 410. In this diagram, the errors distributed to pixels within the horizontal window 420 are classified as firstly, errors distributed directly from pixels of previous scanlines 422, and secondly, errors distributed directly from pixels of the current scanline 424. Previously processed pixels 400 are depicted as shaded. Errors distributed directly from pixels of previous scanlines 422 to pixels of the horizontal window 420 are shown by bold downward pointing arrows 402-404. Errors distributed directly from pixels of the current scanline 424 to pixels of the horizontal window 420 are shown by bold right pointing arrows 406-408. The neighbourhood error of a current pixel 410 is the sum of:

(a) the errors distributed directly to the pixel 410 from pixels of previous scanlines 422, and

(b) the errors distributed directly to the pixel 410 from pixels of the current scanline 424.

The disclosed modification to standard error diffusion uses the errors distributed directly to pixels in the horizontal window 420 around the current pixel 410 to calculate a modified neighbourhood error value for the current pixel 410. The un-modified neighbourhood error value is still calculated in the usual way. The modified

neighbourhood error value is used instead of the unmodified neighbourhood error value in the standard error diffusion pixel decision rule to determine the output value for the current pixel 410. The error for the current pixel 410 is then determined as for standard error diffusion using an input grey value for the current pixel 410, its un-modified neighbourhood error and its halftone output value. The extent 426 of the horizontal window 420 used to calculate a modified neighbourhood error value for a current pixel 410 is determined by the input grey value of the current pixel 410. Because errors distributed to pixels in the horizontal window 420 around the current pixel 410 contribute to the pixel decision rule, the name of "spread decision error diffusion" has been given to this modification to error diffusion. The disclosed modification to standard error diffusion does not require any additional line store memories.

Many methods are possible for determining a modified neighbourhood error value for the current pixel 410 from errors distributed to pixels of the horizontal window 420 around the current pixel 410.

One approach, used in the preferred embodiment described below, is to compute the modified neighbourhood error for the current pixel 410 as a function of:

1) the neighbourhood errors of pixels within the horizontal window 420 which have already been processed;

2) the neighbourhood error of the current pixel 410 and

3) the estimated neighbourhood errors of pixels within the horizontal window 420 which are yet to be processed.

The neighbourhood error of a yet to be processed pixel of the horizontal window 420 can be estimated by adding together

1) the total error distributed directly to the pixel from pixels of the previous scanline 422, and

2) the error distributed directly to the *current* pixel 410 from the pixels of the current scanline 422.

One method of computing the modified neighbourhood error is to compute the modified neighbourhood error as the minimum value of the neighbourhood errors of

pixels within the horizontal window 420 when the current pixel 410 has a low grey value, and, as the maximum of the neighbourhood errors of pixels within the horizontal window 420 when the current pixel 410 has a high grey value. This method has been found to be effective in removing worm artifacts.

It has also been found that a reduction in processing can be obtained while still achieving the goal of removing worm artifacts. This is achieved as follows. When the current pixel has a low grey value, the modified neighbourhood error is computed as the minimum value of the neighbourhood error of the current pixel 410 and the neighbourhood errors of pixels (416, 418) at the extremities of the horizontal window 420. When the current pixel has a high grey value, the modified neighbourhood error is computed as the maximum of the neighbourhood error of the current pixel 410 and the neighbourhood errors of pixels (416, 418) at the extremities of the horizontal window 420. This is the method described in detail in the preferred embodiment below.

Fig. 5 shows a halftone pattern 500 generated by standard error diffusion modified according to the preferred embodiment for an input image region of a constant value of 250 (on a magnitude scale of 0-255). This halftone pattern should be compared with that of Fig. 2. The halftone pattern of Fig. 5 has no worm artifacts as can be seen by examining an indicative inset 502, in which the dots are well separated from each other.

Fig. 6 shows a method of computing the modified neighbourhood error of the preferred embodiment. The pixels 608, 612 at the extremities of the horizontal window around the current pixel 610 are defined by 2 parameters, a pixel displacement ahead of the current pixel, d_lead (ie 604), and a pixel displacement behind the current pixel, d_lag (ie 602). These 2 parameters, d_lead and d_lag are determined according to the input grey value of the current pixel 610.

The neighborhood error of the current pixel 610 at a position (x,y) is given by:

$$e\_nbr\_curr_{x,y}+e\_nbr\_prev_{x,y} \qquad [1]$$

where:

$e\_nbr\_curr_{x,y}$ is the error distributed directly to $pixel_{x,y}$ from pixels of the current scanline;

$e\_nbr\_prev_{x,y}$ is the error distributed directly to $pixel_{x,y}$ from pixels of previous scanlines;

x represents a pixel position number along a scanline, increasing to the right; and

y represents a scanline number, increasing downwards, as depicted by axes 616

The neighborhood error of a pixel 608 further back along the scanline 618 is given by:

$$e\_nbr\_curr_{x-d\_lag,y}+e\_nbr\_prev_{x-d\_lag,y} \quad [2]$$

The estimated neighborhood error of a pixel 612 further forward along the scanline 618 is given by:

$$e\_nbr\_curr_{x,y}+e\_nbr\_prev_{x+d\_lead,y} \quad [3]$$

The shaded horizontal arrow 620 directed to pixel 612 at position (x+d_lead,y) represents an estimate of $e\_nbr\_curr_{x+d\_lead,y}$. This estimate is provided by the value of $e\_nbr\_curr_{x,y}$.

The modified neighborhood error value for the current pixel 610 is the minimum or maximum of the three values [1], [2], and [3].

The preferred embodiment is described for the case of halftoning a monochrome 8 bit per pixel input image to a bi-level output image. A monochrome input image with pixel values $i_{x,y}$ being integers in the range 0 to 255 is used to generate a bi-level halftone output image with pixel values $o_{x,y}$ being the integers 0 and 255.

Fig. 7 provides a block diagram representation of a prior art implementation of Floyd Steinberg error diffusion. The figure describes the processing performed per pixel, to calculate a halftone output value for the pixel from a pixel input value and stored error values. The same processing cycle is repetitively applied, to process a scanline of pixels.

The halftone output value for a pixel at position (x, y) (this pixel being designated as $pixel_{x,y}$) is determined according to a pixel decision rule defined by the following equation:

$$\text{if } i_{x,y} + e\_nbr_{x,y} > th \text{ then } o_{x,y} = 255; \text{ else } o_{x,y} = 0 \qquad [4]$$

where:

$i_{x,y}$ is an input image value of the $pixel_{x,y}$, this being provided on a line 800;

$e\_nbr_{x,y}$ is a neighbourhood error value at the $pixel_{x,y}$, this being provided on a line 854, and being the sum of errors distributed to the $pixel_{x,y}$;

th is a threshold value implicit in a threshold process 806; and

$o_{x,y}$ is the halftone output value of the $pixel_{x,y}$ (0 or 255) provided on an output line 810.

The input image value $i_{x,y}$ is provided on a line 800 to an adder process 802 which also receives the neighborhood error value $e\_nbr_{x,y}$ on the line 854. The adder process 802 outputs a sum of the aforementioned parameters on a line 804 which is directed to the threshold process 806. The threshold process 806 performs the threshold decision described by the equation [4], producing the halftone output value $o_{x,y}$ of the $pixel_{x,y}$ on a line 810.

The error for the $pixel_{x,y}$ at position (x, y) is determined according to the following equation:

$$e_{x,y} = i_{x,y} + e\_nbr_{x,y} - o_{x,y} \qquad [5]$$

where:

$e_{x,y}$ is the error at the $pixel_{x,y}$ at position (x, y).

The sum of the image input value $i_{x,y}$ and neighborhood error value $e\_nbr_{x,y}$ is directed to a subtraction process 816 on a line 808. The halftone output value $o_{x,y}$ of the

$pixel_{x,y}$ is similarly directed to the subtraction process 816 by a line 812, whereupon the subtraction process 816 produces error $e_{x,y}$ on a line 814.

The error distributions from the current $pixel_{x,y}$ at position (x, y) to neighbouring pixels, at pixel locations (x+1,y), (x+1,y+1), (x,y+1), (x-1,y+1) being $eA_{x,y}$, $eB_{x,y}$, $eC_{x,y}$ and $eD_{x,y}$ respectively, are retrieved from an error distribution look up table 820 using $e_{x,y}$ as an index value on the line 814. Accordingly, $eA_{x,y}$, $eB_{x,y}$, $eC_{x,y}$ and $eD_{x,y}$ are output from the table 820 on lines 818, 840, 830 and 828 respectively. The values in the error distribution table 820 are pre-computed to be values consistent with the error weighting coefficients of the error diffusion mask which is shown in Fig. 1. The labels A, B, C, D refer to the 4 pixel positions around a current pixel to which current pixel error is distributed, this relationship being depicted in an inset 826.

An error sum, $e\_nbr\_prev_{x-1,y+1}$ defined by the following equation and being a sum of error distributions from the current and 2 previous pixels of the current scanline is written to the error line store memory 848.

$$e\_nbr\_prev_{x-1,\,y+1} = eB_{x-2,\,y} + eC_{x-1,\,y} + eD_{x,\,y} \qquad [6]$$

where:

$e\_nbr\_prev_{x,\,y}$ is the error distributed directly to the $pixel_{x,\,y}$ at position (x, y) from pixels of previous scanlines. Considering Fig. 7, $eB_{x,\,y}$ is provided from the table 820 on a line 840 and directed to a latch 842. The latch 842 introduces a delay, and thus outputs $eB_{x-1,\,y}$ on a line 844 which is directed to an adder process 846. $eC_{x,\,y}$ is provided from the table 820 on a line 830, which is also directed to the adder process 846. A sum of $eB_{x-1,\,y}$ and $eC_{x,\,y}$ is accordingly output from the adder process 846 to a latch 834. The latch 834 introduces a delay, and thus outputs $eB_{x-2,\,y} + eC_{x-1,\,y}$, directing this sum to an adder 836. $eD_{x,y}$ is provided from the table 820 on a line 828 which is also directed to the adder process 836. The adder process 836 thus outputs the quantity on the right hand side of the equation [6], ie $e\_nbr\_prev_{x-1,\,y+1}$, on a line 838, directed to the line store memory location 862, into which it is written.

Accordingly, when $pixel_{x,y}$ is processed, the error sum $e\_nbr\_prev_{x-1,\ y+1}$ is computed and written to the error line store 848. This error sum is the error distributed directly to pixel (x-1, y+1) from the pixels of scanline y.

The neighbourhood error for $pixel_{x,y}$ is determined by adding an error distribution from the previous pixel of the current scanline and an error sum read from the error line store memory as described by the following equation:

$$e\_nbr_{x,\ y} = eA_{x-1,\ y} + e\_nbr\_prev_{x,\ y} \qquad [7]$$

Accordingly, an error distribution $eA_{x,\ y}$ from the previous pixel (x-1) of the current scanline (y) is provided on a line 818 and directed to a latch 852. The latch 852 introduces a delay, outputting $eA_{x-1,\ y}$, and directing this to an adder process 858. Also directed to this adder process 858 is an error sum $e\_nbr\_prev_{x,\ y}$ which is read from a memory location 850 in the line store 848, and provided to the adder process 858 on a line 860. The adder process outputs the left hand side of the equation [7] ie $e\_nbr_{x,\ y}$, and directs this on a line 854 to the adder process 802.

Fig. 8 provides a physical interpretation of equation [6]. Considering equation [6] in conjunction with Fig. 7, the error sum, $e\_nbr\_prev_{x-1,y+1}$ being a sum of error distributions from the current and 2 previous pixels of the current scanline is written to the error line store memory on the line 838. Fig. 8 depicts a rectangular pixel grid 700, a current $pixel_{x,y}$ being designated by "x,y" in a top left corner of pixel position 702. The error distributions from the current $pixel_{x,y}$ to neighboring pixels 704, 706, 708, and 710 are indicated by the parameters $A_{x,y}$, $B_{x,y}$, $C_{x,y}$ and $D_{x,y}$ in those pixel positions. The aforementioned error sum $e\_nbr\_prev_{x-1,y+1}$ as represented by the right hand side of equation [6] ie $eB_{x-2,\ y} + eC_{x-1,\ y}$ and $eD_{x,\ y}$ is seen to comprise contributions from the current pixel 702, and the previous two pixels 718, and 720, these contributions being depicted by arrows 712, 714 and 716 respectively. Similarly, the term $e\_nbr\_prev_{x,\ y}$ in equation [7] ie $eB_{x-1,\ y-1} + eC_{x,\ y-1}$ and $eD_{x+1,\ y-1}$ is seen to comprise contributions from pixel 722, and the previous two pixels 724, and 726, these contributions being depicted by

arrows 728, 730 and 732 respectively. Furthermore, with the additional contribution from pixel 718 on the same scanline as $pixel_{x,y}$ ie $A_{x-1,y}$ as depicted by a dashed arrow 734, the left hand term of equation [7] is provided.

Fig. 9 shows a block diagram for the preferred embodiment, and depicts processing performed per pixel, to calculate a halftone output value for a pixel from the pixel input value and stored error values. The same processing cycle is repetitively applied to process a scanline of pixels.

The halftone output value for a $pixel_{x,y}$ at position (x, y) is determined according to a pixel decision rule which is the same as that of the implementation of Floyd Steinberg error diffusion (ie equation [4]), except that a modified neighbourhood error value is used:

$$i_{x,y} + e\_nbr\_mod_{x,y} > th \text{ then } o_{x,y} = 255; \text{ else } o_{x,y} = 0 \qquad [8]$$

where;

$e\_nbr\_mod_{x,y}$ is a modified neighbourhood error at the $pixel_{x,y}$, defined below.

An input image value $i_{x,y}$ is provided on a line 900 to an adder process 902 which also receives the modified neighborhood error value $e\_nbr\_mod_{x,y}$ on the line 954. The adder process 902 outputs a sum of the aforementioned parameters on a line 904 which is directed to a threshold process 906. The threshold process 906 performs the threshold decision described by the equation [8], producing the halftone output value $o_{x,y}$ of the $pixel_{x,y}$ on a line 908. An error $e_{x,y}$ for the current $pixel_{x,y}$ is produced on a line 920, and directed to a lookup table 922. The error distributions from the current $pixel_{x,y}$ at position (x, y) to neighbouring pixels, at pixel locations (x+1,y), (x+1,y+1), (x,y+1), (x-1,y+1) being $eA_{x,y}$, $eB_{x,y}$, $eC_{x,y}$ and $eD_{x,y}$ respectively, are retrieved from an error distribution look up table 922 using $e_{x,y}$ as an index value on the line 920. Accordingly, $eA_{x,y}$, $eB_{x,y}$, $eC_{x,y}$ and $eD_{x,y}$ are output from the table 922 on lines 926, 928, 936, and 942 respectively. The values in the error distribution table 922 are pre-computed to be values consistent with the error weighting coefficients of the Floyd Steinberg error diffusion mask which

are shown in Fig. 1. The labels A, B, C, D refer to the 4 pixel positions around a current pixel to which current pixel error is distributed, this relationship being depicted in an inset 924.

An error sum, $e\_nbr\_prev_{x-1,y+1}$ defined by the following equation and being a sum of error distributions from the current and 2 previous pixels of the current scanline is written to the error line store memory 956, in a memory position 960.

$$e\_nbr\_prev_{x-1,\,y+1} = eB_{x-2,\,y} + eC_{x-1,\,y} \text{ and } eD_{x,\,y} \qquad [9]$$

where:

$e\_nbr\_prev_{x,\,y}$ is the error distributed directly to the $pixel_{x,\,y}$ at position (x, y) from pixels of previous scanlines. Considering Fig. 9, $eB_{x,\,y}$ is provided from the table 922 on a line 928 and directed to a latch 930. The latch 930 introduces a delay, and thus outputs $eB_{x-1,\,y}$ on a line 932 which is directed to an adder process 934. $eC_{x,\,y}$ is provided from the table 922 on a line 936, which is also directed to the adder process 934. A sum of $eB_{x-1,\,y}$ and $eC_{x,\,y}$ is accordingly output from the adder process 934 to a latch 940. The latch 940 introduces a delay, and thus outputs $eB_{x-2,\,y} + eC_{x-1,\,y}$, directing this sum to an adder 944. $eD_{x,y}$ is provided from the table 922 on a line 942 which is also directed to the adder process 944. The adder process 944 thus outputs the quantity on the right hand side of the equation [9], ie $e\_nbr\_prev_{x-1,\,y+1}$, on a line 948, directed to the line store memory location 960, into which it is written.

It is seen that processing within a dashed box 911 in Fig. 9 is substantially the same as that described in relation to Fig. 7. The error for the $pixel_{x,y}$ at position (x, y) is determined the same way as in the implementation of Floyd Steinberg error diffusion, using the un-modified neighbourhood error at the current pixel, $e\_nbr_{x,y}$. Accordingly, the image input value $i_{x,y}$ (present on a line 913) and a neighborhood error value $e\_nbr_{x,y}$ (present on a line 915) are directed to an adder process 914, whose output is directed to a subtraction process 918 by a line 916. The halftone output value $o_{x,y}$ of the $pixel_{x,y}$ is similarly directed to the subtraction process 918 by a line 910, whereupon the subtraction

process 918 produces error $e_{x,y}$ on the line 920. Both the un-modified neighbourhood error $e\_nbr_{x,y}$ at the current $pixel_{x,y}$ and the modified neighbourhood error $e\_nbr\_mod_{x,y}$ at the current $pixel_{x,y}$ are computed using values read from the error line store 956 and error distributions from previous pixels of the current scanline as is now explained.

Additional memory is required for the implementation of the modification to spread decision error diffusion. This is shown as a lead error buffer 970 and a lag error buffer 903. The lead error buffer 970 stores sums of errors distributed directly to pixels of the current scanline from previous scanlines. The lag error buffer 903 stores the un-modified neighbourhood error values of previously processed pixels of the current scanline. Data can be read from the lead error buffer 970 from a memory location 986. The address of the memory location 986 can vary, as depicted by the bilateral arrow spanning the memory location 986. This address is determined by address data on a line 988, which is derived from a lead displacement table 992. The lead displacement table 992 outputs address data on the line 988 dependent upon the value of the input image value $i_{x,y}$ on a line 994. Similarly, data can be read from the lag error buffer 903 from a memory location 901. The address of the memory location 901 can vary, as depicted by the bilateral arrow spanning the memory location 901. This address is determined by address data on a line 921, which is derived from a lag displacement table 909. The lag displacement table 909 outputs address data on the line 921 dependent upon the value of the input image value $i_{x,y}$ on a line 905.

The size of this additional memory is however small compared to the additional line store memory required by the previously mentioned methods for preventing worm artifacts, because the lead error buffer 970 and the lag error buffer 903 are only required for storing error values of pixels on the current scanline in a small window (420, see Fig. 4) around the current pixel. In the processing cycle for $pixel_{x,y}$ at position (x, y) an error sum, $e\_nbr\_prev_{x + d\_max,\ y}$ is read on a line 974 from the error line store 956 to the lead error buffer 970. d_max is a maximum displacement value which is greater than or equal to any of the displacement values in the lead displacement table and the lag displacement table. An error sum, $e\_nbr\_prev_{x,\ y}$ is read on a line 982 from the lead error buffer 970.

In this respect, the lead error buffer 970 operates as a First In First Out buffer. This error sum value $e\_nbr\_prev_{x,\ y}$ on the line 982 is used to calculate the un-modified neighbourhood error $e\_nbr_{x,\ y}$ on a line 917 for $pixel_{x,y}$ at position (x, y), in the same way as for the standard error diffusion.

An error sum $e\_nbr\_prev_{x\ +\ d\_lead,\ y}$ is read on a line 968 from the memory location 986 in the lead error buffer 970, using a lead displacement, d_lead, in the form of an address on the line 988, this being read from the lead displacement table 992 using the current pixel input image value, $i_{x,\ y}$, as index on a line 994. This error sum value on the lines 968 is used to calculate an estimated error value for pixel (x + d_lead, y) according to the equation:

$$estimate(e\_nbr_{x\ +\ d\_lead,\ y}) = eA_{x-1,\ y} + e\_nbr\_prev_{x\ +\ d\_lead,\ y} \qquad [10]$$

Equation [10] is implemented by providing $e\_nbr\_prev_{x\ +\ d\_lead,\ y}$ on the line 968 to an adder process 919, and also providing $eA_{x-1,\ y}$ on a line 962, to the adder process 919, which produces $estimate(e\_nbr_{x\ +\ d\_lead,\ y})$ on a line 966. The un-modified neighbourhood error for $pixel_{x,y}$ at position (x, y), $e\_nbr_{x,\ y}$, is written on a line 998 to the lag error buffer 903 so that it can be retrieved, if required, for the processing of pixels a small way ahead of the current pixel on the same scanline. The un-modified neighbourhood error for pixel (x + d_lag, y), ie $e\_nbr_{x\ +\ d\_lag,\ y}$, is read from a memory position 901 in the lag error buffer 903 on a line 996. This is performed using a lag displacement d_lag, on an address line 921 read from the lag displacement table 909 using the current pixel input image value, $i_{x,\ y}$, on a line 905 as an index.

The modified neighbourhood error at the current $pixel_{x,y}$ is calculated in a min/max process 976 as either the minimum or the maximum of the following three neighbourhood error values:

$estimate(e\_nbr_{x\ +\ d\_lead,\ y})$

$e\_nbr_{x,\ y}$

$e\_nbr_{x - d\_lag, y}$

these values being present on lines 966, 917, and 996 respectively.

The calculation performed by the min/max process 976 is a minimum operation when the current pixel input image value, $i_{x, y}$, is low (<= 127) and is a maximum operation when the current pixel input image value is high (>= 128), where the current pixel input image value, $i_{x, y}$ is present on a line 978.

The following table of lead and lag pixel displacement values are found to be suitable for bi-level error diffusion halftoning of 8 bit per pixel monochrome images where scanline processing is left to right for all scanlines and the Floyd Steinberg error diffusion mask is used.

Input grey values	Lag distance	Lead distance
1, 254	4	7
2, 253	3	5
3, 252	2	4
4 - 6, 249 - 251	1	3
7 - 16, 239 - 248	1	2
17 - 18, 237 - 238	0	1
19 - 31, 224 - 236	0	1
all other grey values	0	0

The modification to error diffusion to remove worm artifacts without requiring the use of additional line store memory disclosed in this patent can be applied to error diffusion with an arbitrary mask extending over an arbitrary number of scanlines. In addition, the modification can be applied to error diffusion for which the scanline processing direction is not simply left to right but includes an arbitrary pattern of left to right and right to left processing. Suitable alternative lead and lag pixel displacement parameters must be determined when the error diffusion mask is altered or when the

scanline processing directions are altered. The modification can also be applied to error diffusion where the output halftoned image has more than 2 levels and where the error diffusion is performed on a colour image.

The method of halftoning an image can be practiced using a conventional general-purpose computer system 1000, such as that shown in Fig. 10, wherein the processes of Fig. 9 may be implemented as software, such as a program executing within the computer system 1000. In particular, the steps of the method of image halftoning are effected by instructions in the software that are carried out by the computer. The software may be divided into two separate parts; one part for carrying out the halftoning methods; and another part to manage the user interface between the latter and the user. The software may be stored in a computer readable medium, including the storage devices described below, for example. The software is loaded into the computer from the computer readable medium, and then executed by the computer. A computer readable medium having such software or computer program recorded on it is a computer program product. The use of the computer program product in the computer effects an advantageous apparatus for image halftoning in accordance with the embodiment of the invention.

The computer system 1000 comprises a computer module 1001, input devices such as a keyboard 1002 and mouse 1003, output devices including a printer 1015 and a display device 1014. A Modulator-Demodulator (Modem) transceiver device 1016 is used by the computer module 1001 for communicating to and from a communications network 1020, for example connectable via a telephone line 1021 or other functional medium. The modem 1016 can be used to obtain access to the Internet, and other network systems, such as a Local Area Network (LAN) or a Wide Area Network (WAN).

The computer module 1001 typically includes at least one processor unit 1005, a memory unit 1006, for example formed from semiconductor random access memory (RAM) and read only memory (ROM), input/output (I/O) interfaces including a video interface 1007, and an I/O interface 1013 for the keyboard 1002 and mouse 1003 and optionally a joystick (not illustrated), and an interface 1008 for the modem 1016. A

storage device 1009 is provided and typically includes a hard disk drive 1010 and a floppy disk drive 1011. A magnetic tape drive (not illustrated) may also be used. A CD-ROM drive 1012 is typically provided as a non-volatile source of data. The components 1005 to 1013 of the computer module 1001, typically communicate via an interconnected bus 1004 and in a manner which results in a conventional mode of operation of the computer system 1000 known to those in the relevant art. Examples of computers on which the embodiments can be practised include IBM-PC's and compatibles, Sun Sparcstations or alike computer systems evolved therefrom.

Typically, the program of the embodiment is resident on the hard disk drive 1010 and read and controlled in its execution by the processor 1005. Intermediate storage of the program and any data fetched from the network 1020 may be accomplished using the semiconductor memory 1006, possibly in concert with the hard disk drive 1010. In some instances, the application program may be supplied to the user encoded on a CD-ROM or floppy disk and read via the corresponding drive 1012 or 1011, or alternatively may be read by the user from the network 1020 via the modem device 1016. Still further, the software can also be loaded into the computer system 1000 from other computer readable medium including magnetic tape, a ROM or integrated circuit, a magneto-optical disk, a radio or infra-red transmission channel between the computer module 1001 and another device, a computer readable card such as a PCMCIA card, and the Internet and Intranets including email transmissions and information recorded on websites and the like. The foregoing is merely exemplary of relevant computer readable mediums. Other computer readable mediums may be practiced without departing from the scope and spirit of the invention.

The method of digital halftoning is preferably implemented in dedicated hardware such as one or more integrated circuits performing the functions or sub functions of halftoning. Such dedicated hardware may include graphic processors, digital signal processors, or one or more microprocessors and associated memories.

Industrial Applicability

It is apparent from the above that the embodiment(s) of the invention are applicable to the computer, data processing and digital image processing industries.

The foregoing describes only one embodiment of the present invention, and modifications and/or changes can be made thereto without departing from the scope and spirit of the invention, the embodiment being illustrative and not restrictive.

In the context of this specification, the word "comprising" means "including principally but not necessarily solely" or "having" or "including" and not "consisting only of". Variations of the word comprising, such as "comprise" and "comprises" have corresponding meanings.

The claims defining the invention are as follows:

1. A method of halftoning an image, said method comprising, for a current pixel, when a current pixel input value is within a range of values, the steps of:

determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and a neighbourhood error value of one or more neighbouring pixels;

determining an output value using a sum of the current pixel input value and the modified neighbourhood error value;

determining an error value as the difference between, firstly, the sum of the current pixel input value and the current pixel neighbourhood error value, and secondly the output value of the current pixel; and

adding proportions of the error value for the current pixel to the neighbourhood error values of as yet unprocessed pixels.

2. A method according to claim 1, wherein said range of values comprises one of a highlight (near white) region and a shadow (near black) region.

3. A method according to claim 1, wherein said one or more neighbouring pixels in the step of determining a modified neighbourhood error value are within a horizontal window about the current pixel.

4. A method according to claim 3, whereby said one or more neighbouring pixels comprise all pixels in said window.

5. A method according to claim 3, wherein said one or more neighbouring pixels comprise pixels at the extremities of said window.

6. A method according to claim 3, wherein an extent of said window is dependent upon a value of the current pixel.

7. A method according to claim 3, wherein said modified neighbourhood error value for said current pixel is dependent upon neighbourhood errors of:

already processed pixels in said window; and

a neighbourhood error of said current pixel; and

estimated neighbourhood errors of as yet unprocessed pixels in said window.

8. A method according to claim 7, wherein an estimated neighbourhood error of an as yet unprocessed pixel in said window is dependent upon:

a total error distributed directly to the unprocessed pixel from pixels of a previous scanline; and

an error distributed directly to the current pixel from pixels of a current scanline.

9. A method according to claim 6, wherein said extent of said window comprises a leading dimension and a lagging dimension, each said dimension being dependent upon the value of the current pixel.

10. A method according to claim 3, wherein for said current pixel having a low grey value, said modified neighbourhood error is a minimum value of neighbourhood errors within said window.

11. A method according to claim 3, wherein for said current pixel having a high grey value, said modified neighbourhood error is a maximum value of neighbourhood errors within said window.

12. A method of halftoning an image, said image comprising a plurality of pixels each having an input value and an assignable output value that can take on one of at least two output values, wherein processing for each pixel comprises the steps of:

determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and, when the pixel input value is within a critical range of values, further using at least one neighbourhood error value of a neighbouring pixel;

determining the output value of a current pixel using a sum of the input value of the current pixel and the modified neighbourhood error value for the pixel;

determining an error value for the current pixel as the difference between, firstly, the sum of the input value of the current pixel and the neighbourhood error value for the pixel, and secondly the output value of the pixel; and

adding proportions of the error value for the current pixel to the neighbourhood error values of yet to be processed neighbouring pixels.

13. A method as claimed in claim 12, wherein the output value of a current pixel is determined by comparison of the sum of the input value of the current pixel and the modified neighbourhood error value for the pixel against a threshold value.

14. A method as claimed in either one of claims 1 and 2, wherein the modified neighbourhood error value for the current pixel is determined using the neighbourhood errors of pixels within a set of pixels positioned around the current pixel, and where the extent of the set of pixels is determined by the current pixel input value.

15. A method as claimed in claim 14, wherein pixels of a scanline are processed one at a time either left to right or right to left, and scanlines are processed one at a time from the top of the image to the bottom of the image.

16. A method as claimed in claim 15, wherein the modified neighbourhood error value for the current pixel is determined using the neighbourhood error value of pixels of the scanline of the current pixel and is determined without directly using the neighbourhood error value of pixels of other scanlines.

17. An apparatus adapted for halftoning an image, said apparatus comprising:

first determining means for determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and a neighbourhood error value of one or more neighbouring pixels;

second determining means for determining an output value using a sum of the current pixel input value and the modified neighbourhood error value;

third determining means for determining an error value as the difference between, firstly, the sum of the current pixel input value and the current pixel neighbourhood error value, and secondly the output value of the current pixel; and

adding means for adding proportions of the error value for the current pixel to the neighbourhood error values of as yet unprocessed pixels.

18. An apparatus according to claim 17, wherein said range of values comprises one of a highlight (near white) region and a shadow (near black) region.

19. An apparatus according to claim 17, wherein said one or more neighbouring pixels in the step of determining a modified neighbourhood error value are within a horizontal window about the current pixel.

20. An apparatus according to claim 19, wherein said one or more neighbouring pixels comprise all pixels in said window.

21. An apparatus according to claim 19, wherein said one or more neighbouring pixels comprise pixels at the extremities of said window.

22. An apparatus according to claim 19, wherein an extent of said window is dependent upon a value of the current pixel.

23. An apparatus according to claim 19, wherein said modified neighbourhood error value for said current pixel is dependent upon neighbourhood errors of:

already processed pixels in said window; and

a neighbourhood error of said current pixel; and

estimated neighbourhood errors of as yet unprocessed pixels in said window.

24. An apparatus according to claim 23, wherein an estimated neighbourhood error of an as yet unprocessed pixel in said window is dependent upon:

a total error distributed directly to the unprocessed pixel from pixels of a previous scanline; and

an error distributed directly to the current pixel from pixels of a current scanline.

25. An apparatus according to claim 22, wherein said extent of said window comprises a leading dimension and a lagging dimension, each said dimension being dependent upon the value of the current pixel.

26. An apparatus according to claim 19, wherein for said current pixel having a low grey value, said modified neighbourhood error is a minimum value of neighbourhood errors within said window.

27. An apparatus according to claim 19, wherein for said current pixel having a high grey value, said modified neighbourhood error is a maximum value of neighbourhood errors within said window.

28. An apparatus adapted for halftoning an image, said apparatus image comprising:

first determining means for determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and, when the pixel input value is within a critical range of values, further using at least one neighbourhood error value of a neighbouring pixel;

second determining means for determining the output value of a current pixel using a sum of the input value of the current pixel and the modified neighbourhood error value for the pixel;

third determining means for determining an error value for the current pixel as the difference between, firstly, the sum of the input value of the current pixel and the neighbourhood error value for the pixel, and secondly the output value of the pixel; and

adding means for adding proportions of the error value for the current pixel to the neighbourhood error values of yet to be processed neighbouring pixels.

29. An apparatus as claimed in claim 28, wherein said second determining means comprises comparing means for comparing the sum of the input value of the current pixel and the modified neighbourhood error value for the pixel against a threshold value.

30. An apparatus as claimed in either one of claims 28 and 29, wherein the modified neighbourhood error value for the current pixel is determined using the neighbourhood errors of pixels within a set of pixels positioned around the current pixel, and where the extent of the set of pixels is determined by the current pixel input value.

31. An apparatus as claimed in claim 30, wherein pixels of a scanline are processed one at a time either left to right or right to left, and scanlines are processed one at a time from the top of the image to the bottom of the image.

32. An apparatus as claimed in claim 31, wherein the modified neighbourhood error value for the current pixel is determined using the neighbourhood error value of pixels of the scanline of the current pixel and is determined without directly using the neighbourhood error value of pixels of other scanlines.

33. A computer readable memory medium for storing a program for apparatus adapted for halftoning an image, said program comprising, for a current pixel, when a current pixel input value is within a range of values, the steps of:

code for a first determining step for determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and a neighbourhood error value of one or more neighbouring pixels;

code for a second determining step for determining an output value using a sum of the current pixel input value and the modified neighbourhood error value;

code for a third determining step for determining an error value as the difference between, firstly, the sum of the current pixel input value and the current pixel neighbourhood error value, and secondly the output value of the current pixel; and

code for an adding step for adding proportions of the error value for the current pixel to the neighbourhood error values of as yet unprocessed pixels.

34. A computer readable memory medium for storing a program for apparatus adapted for halftoning an image, said image comprising a plurality of pixels each having an input value and an assignable output value that can take on one of at least two output values, wherein said program comprises, for each pixel, the steps of:

code for a first determining step for determining a modified neighbourhood error value for the current pixel using a neighbourhood error value for the current pixel and,

when the pixel input value is within a critical range of values, further using at least one neighbourhood error value of a neighbouring pixel;

code for a second determining step for determining the output value of a current pixel using a sum of the input value of the current pixel and the modified neighbourhood error value for the pixel;

code for a third determining step for determining an error value for the current pixel as the difference between, firstly, the sum of the input value of the current pixel and the neighbourhood error value for the pixel, and secondly the output value of the pixel; and

code for an adding step for adding proportions of the error value for the current pixel to the neighbourhood error values of yet to be processed neighbouring pixels.

35. A method of halftoning an image, substantially as described herein with reference to Figs. 4, 6 and 9, or with reference to Figs 4, 6 and 10.

36. An apparatus adapted for halftoning an image, substantially as described herein with reference to Figs. 4, 6 and 9, or with reference to Figs 4, 6 and 10.

37. A computer readable memory medium for storing a program for apparatus adapted for halftoning an image, substantially as described herein with reference to Figs. 4, 6 and 9, or with reference to Figs 4, 6 and 10.

DATED this Twenty-fourth Day of November, 2000

Canon Kabushiki Kaisha

Patent Attorneys for the Applicant

SPRUSON & FERGUSON

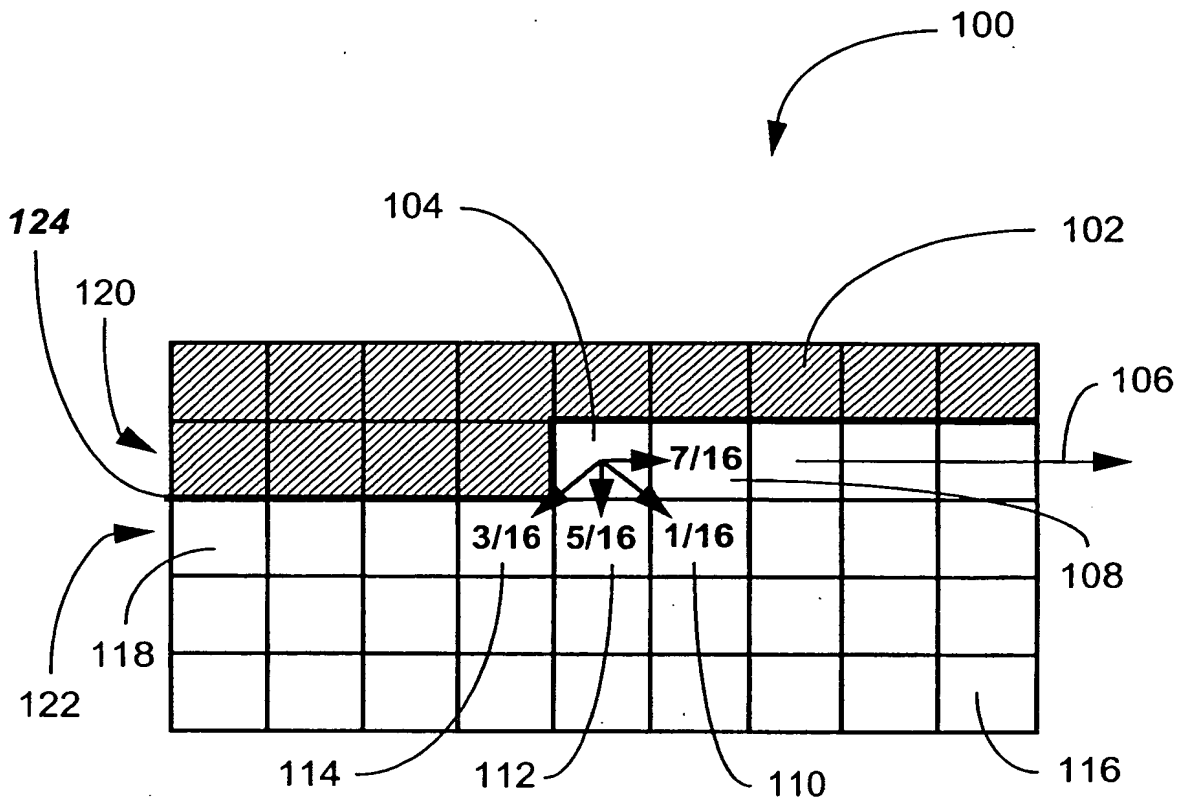

Fig. 1
(prior art)

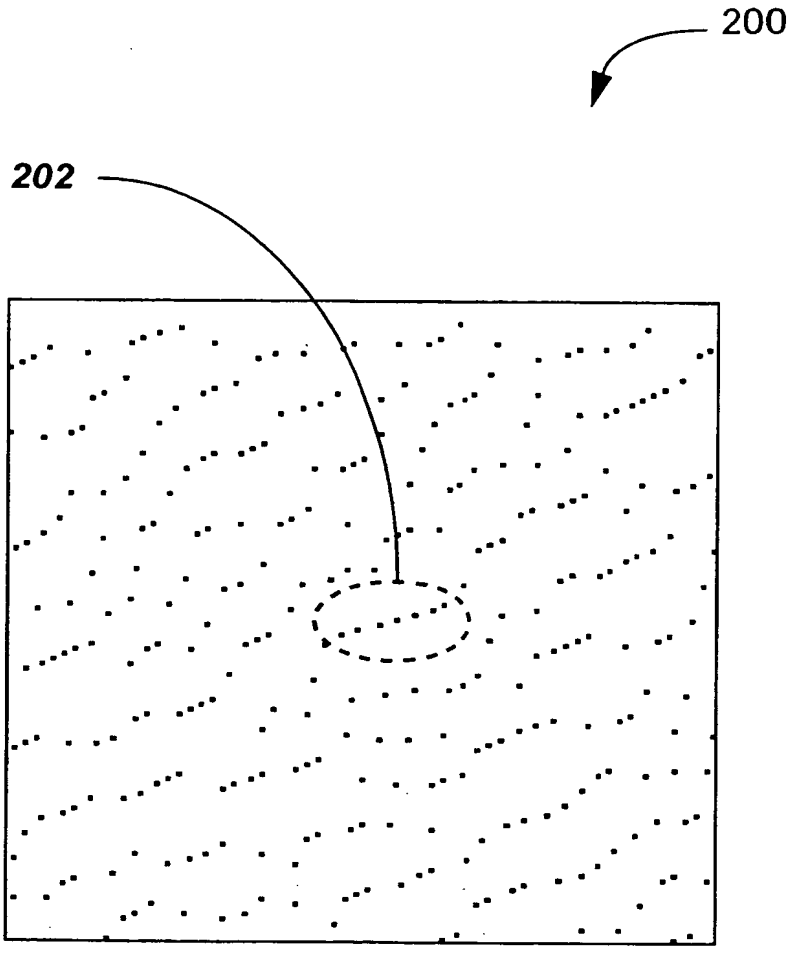

Fig. 2
(prior art)

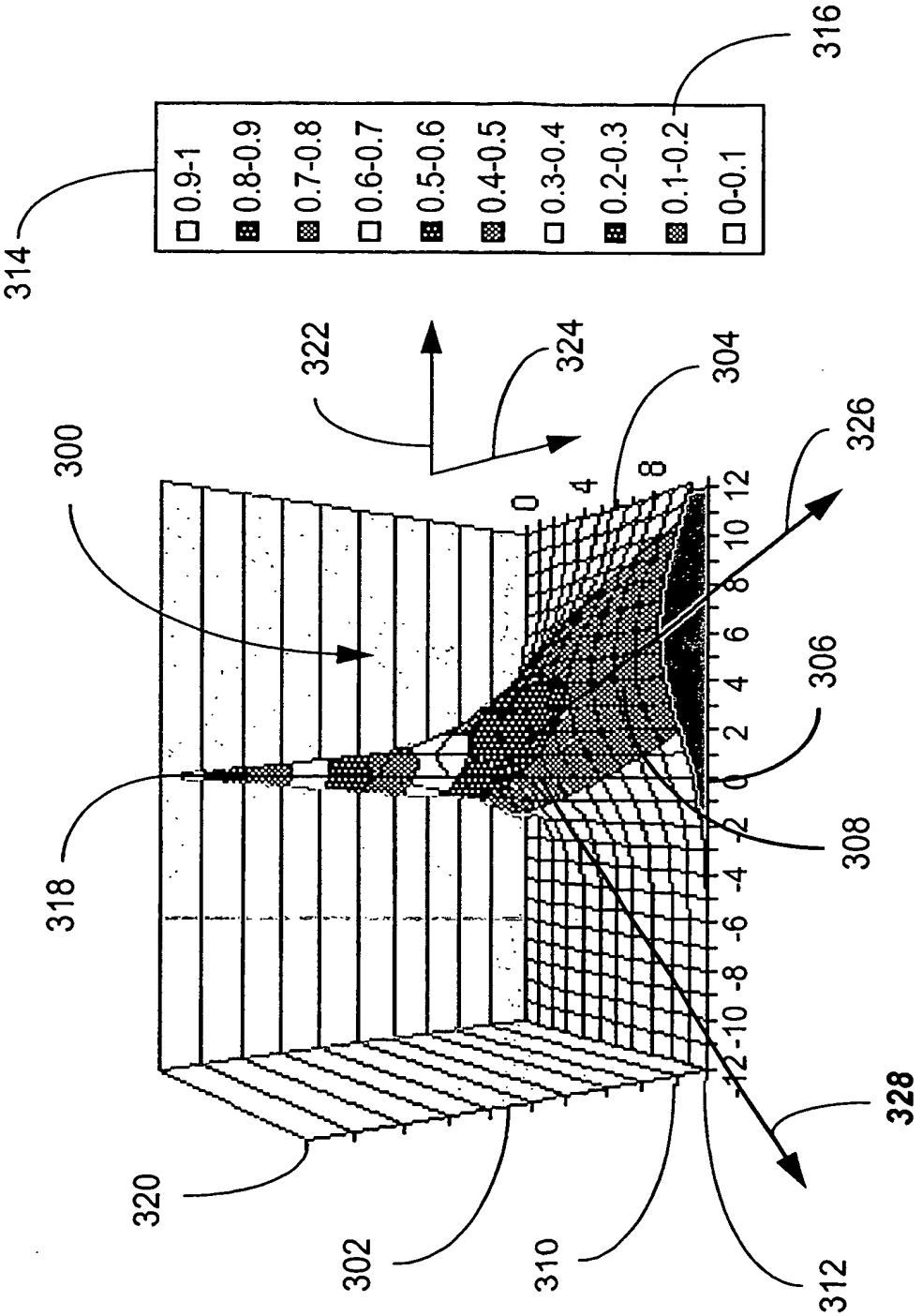
0.9-1
0.8-0.9
0.7-0.8
0.6-0.7
0.5-0.6
0.4-0.5
0.3-0.4
0.2-0.3
0.1-0.2
0-0.1
314
316
322
324
304
300
326
306
318
308
320
328
302
310
312
0
4
8
12
10
8
6
4
2
0
-2
-4
-6
-8
-10
-12

Fig. 3

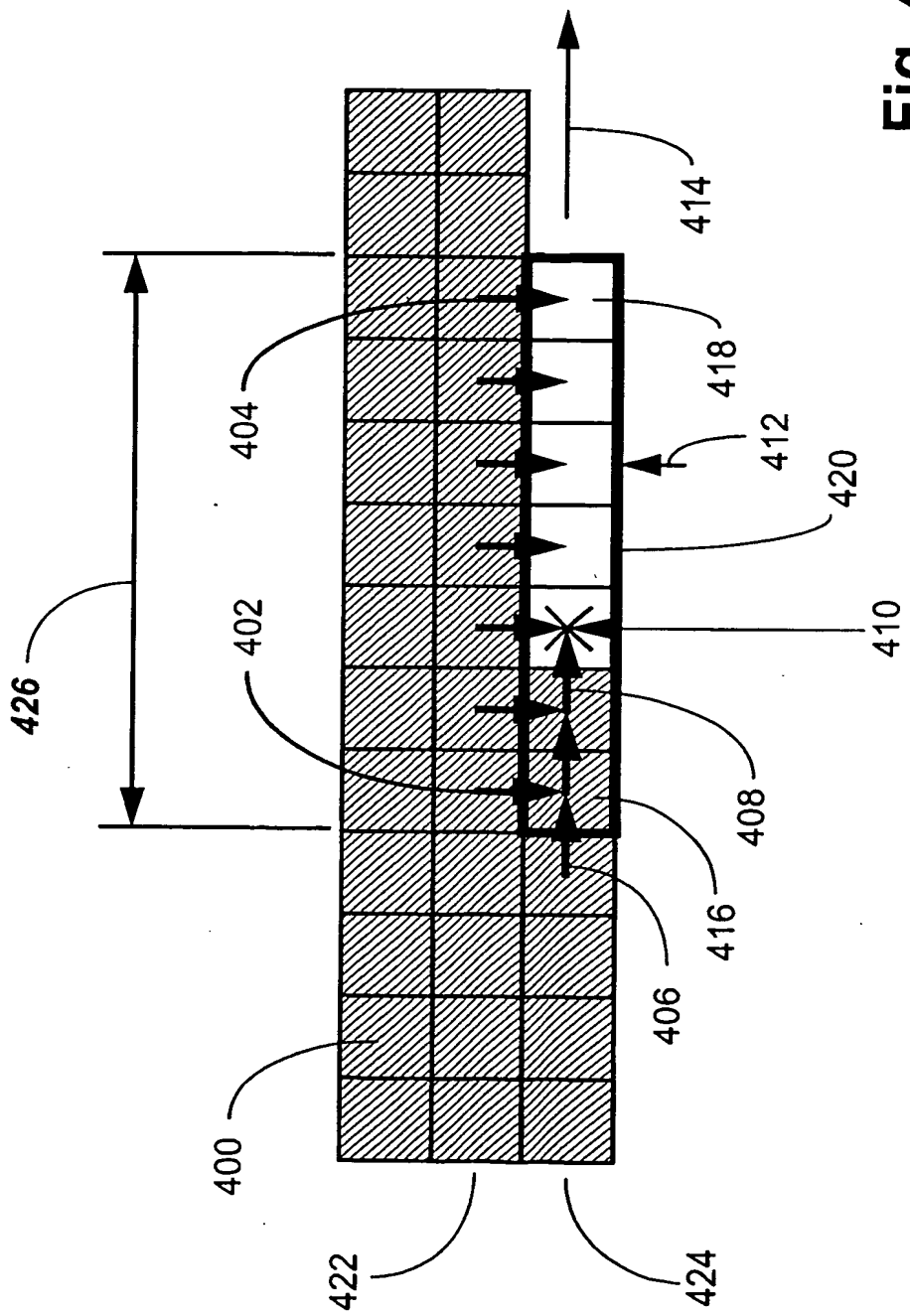

Fig. 4

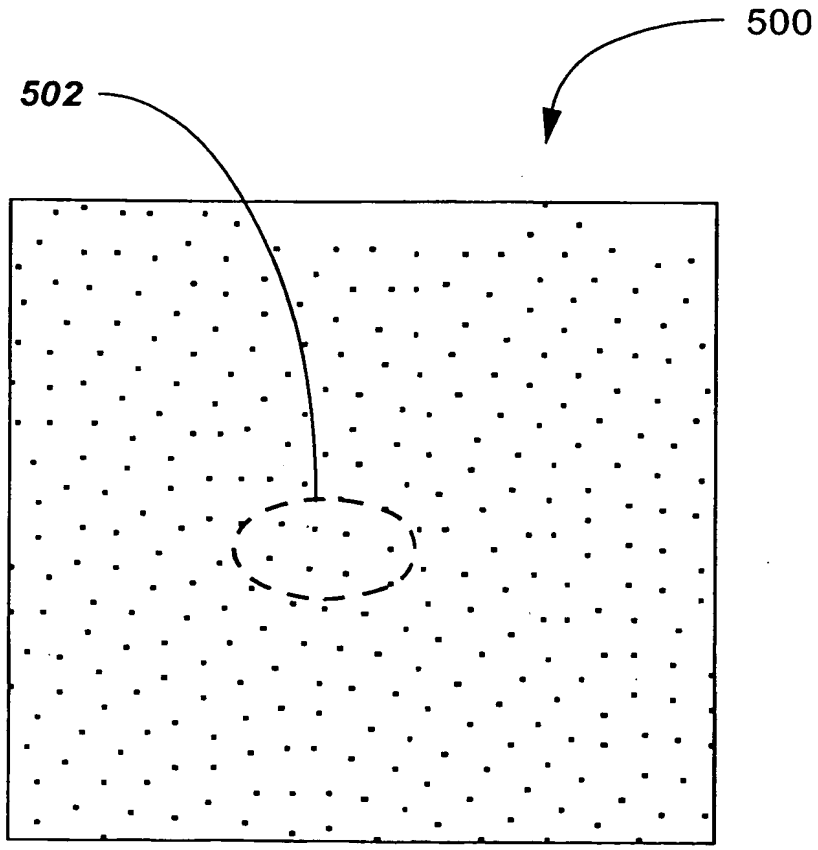

Fig. 5

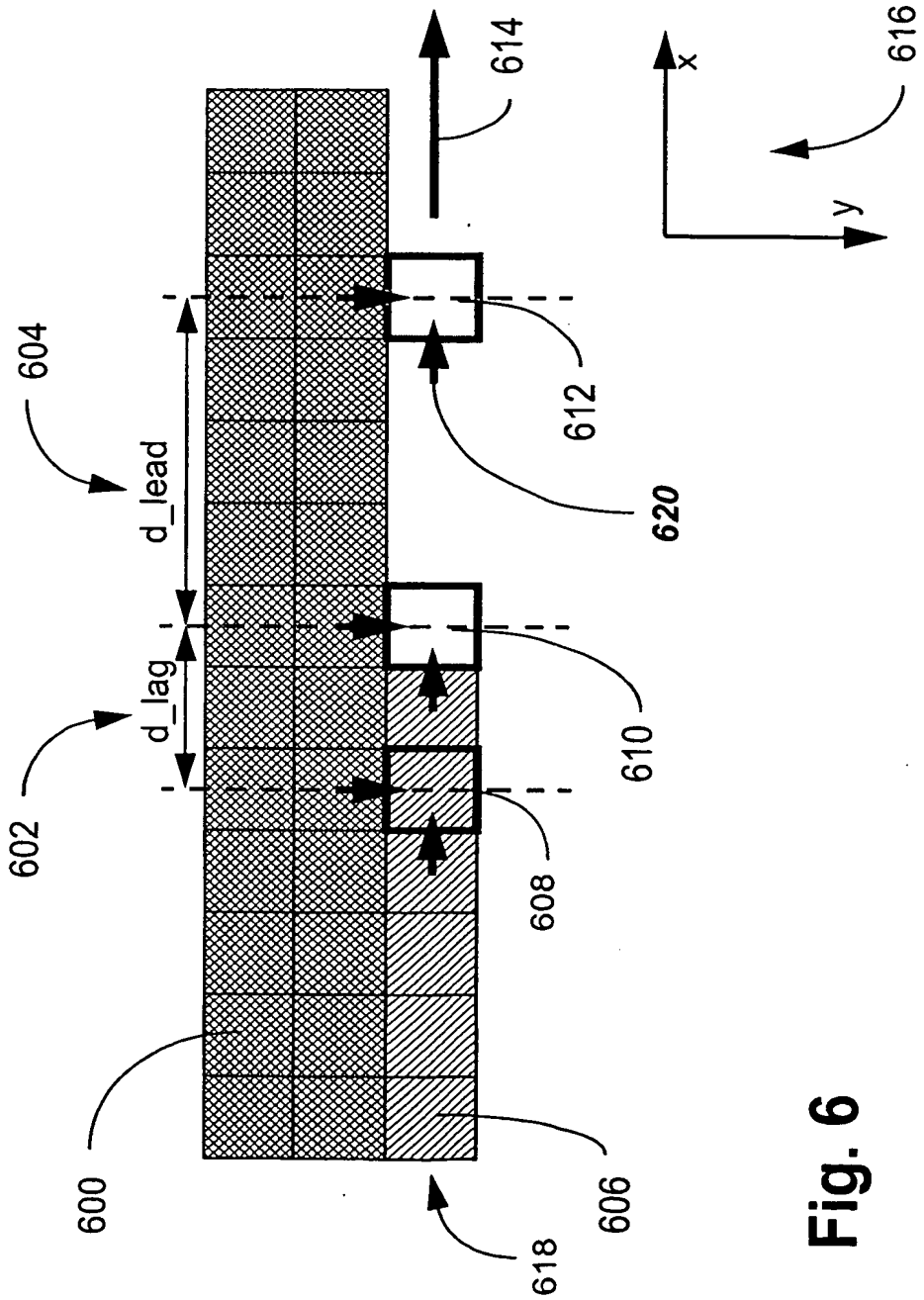

Fig. 6

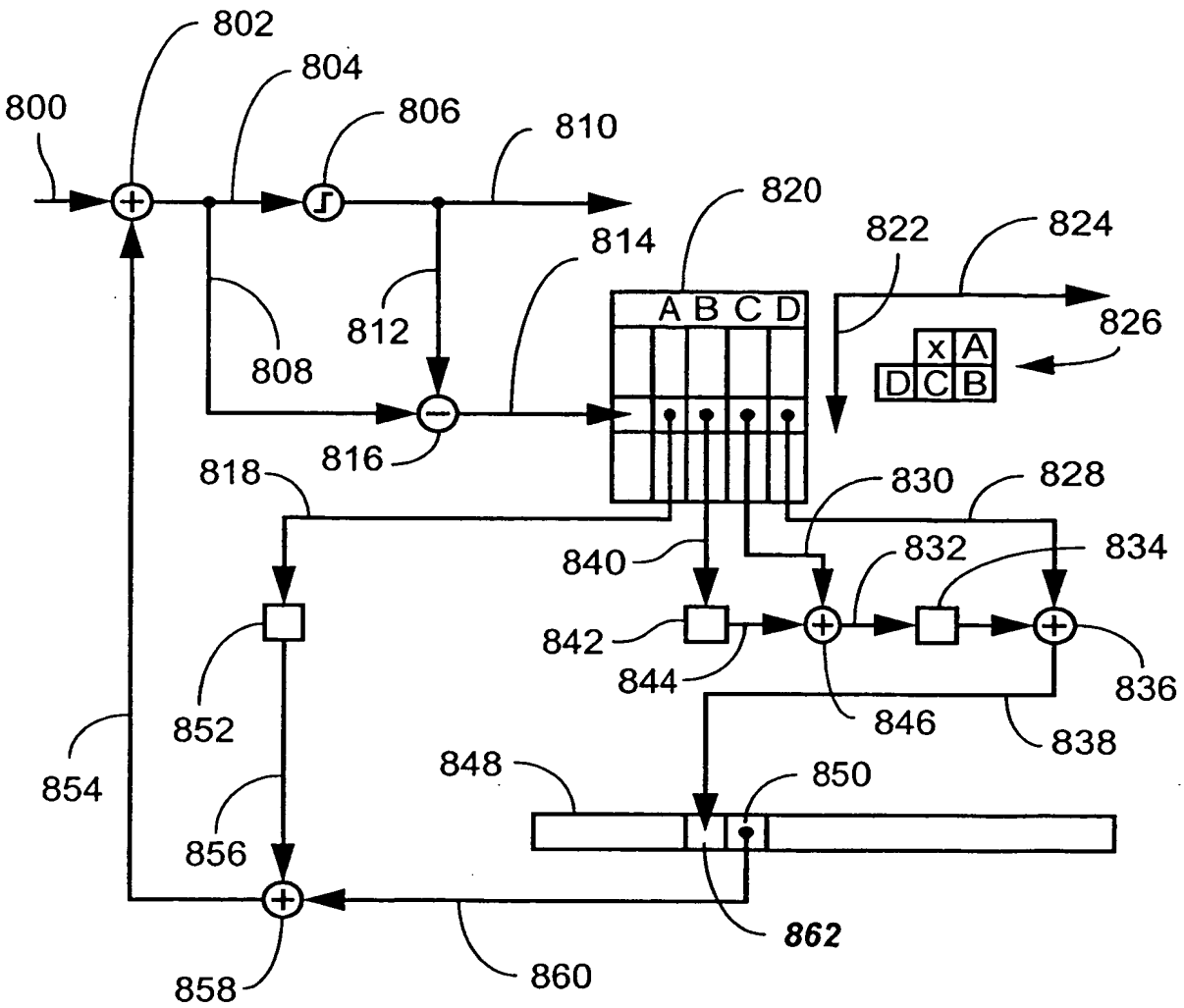

Fig. 7

(prior art)

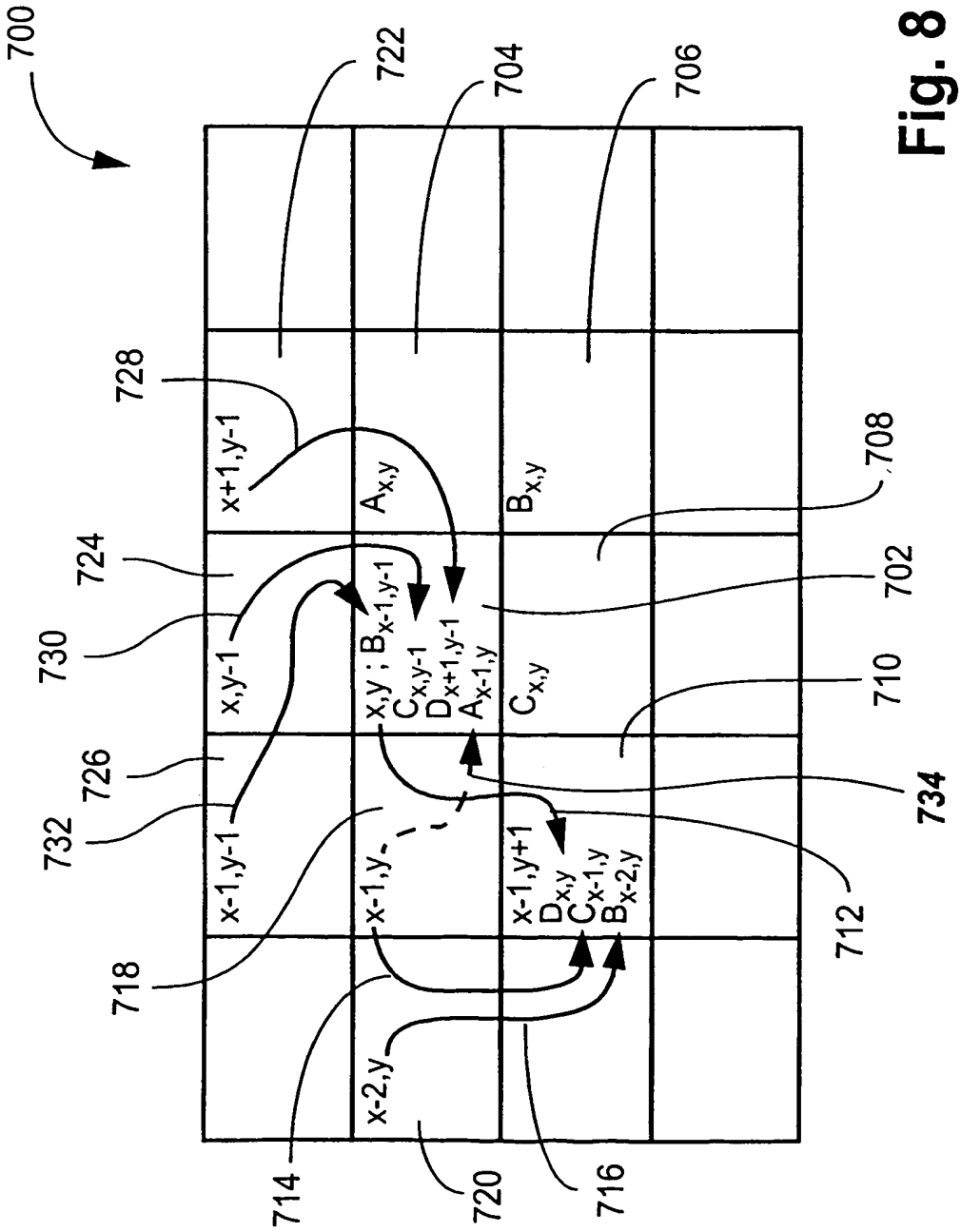
700
722
704
706
728
708
724
702
730
710
726
734
732
712
718
714
720
716
x+1,y-1
A x,y
B x,y
x,y-1
x,y ; B x-1,y-1
C x,y-1
D x+1,y-1
A x-1,y
C x,y
x-1,y-1
x-1,y
x-1,y+1
D x,y
C x-1,y
B x-2,y
x-2,y

Fig. 8

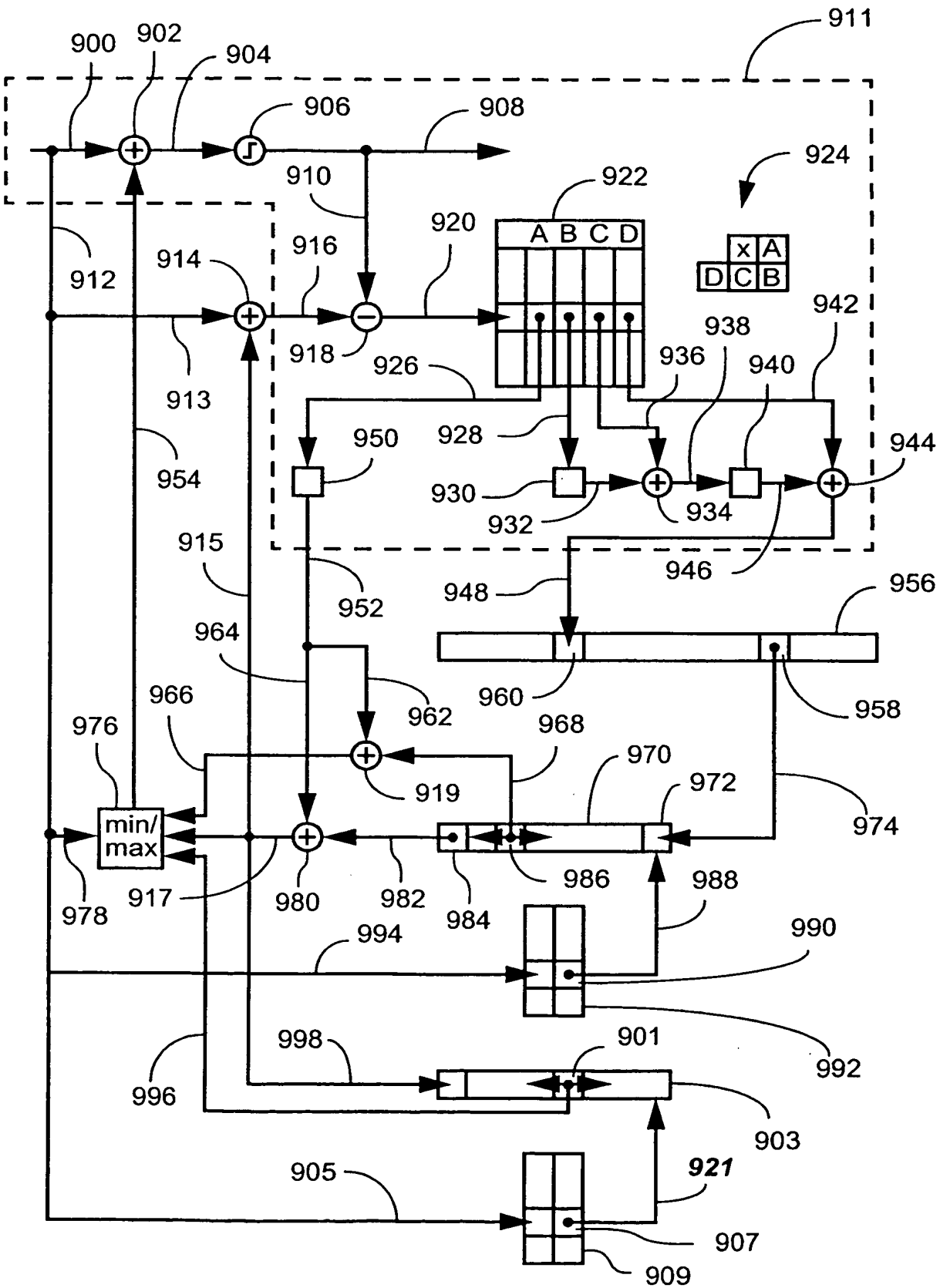
911
900
902
904
906
908
910
912
914
916
918
920
922
924
A B C D
x A
D C B
913
926
928
930
932
934
936
938
940
942
944
946
948
950
954
915
952
956
958
960
962
964
966
968
970
972
974
976
978
min/
max
917
919
980
982
984
986
988
990
992
994
996
998
901
903
905
921
907
909

Fig. 9

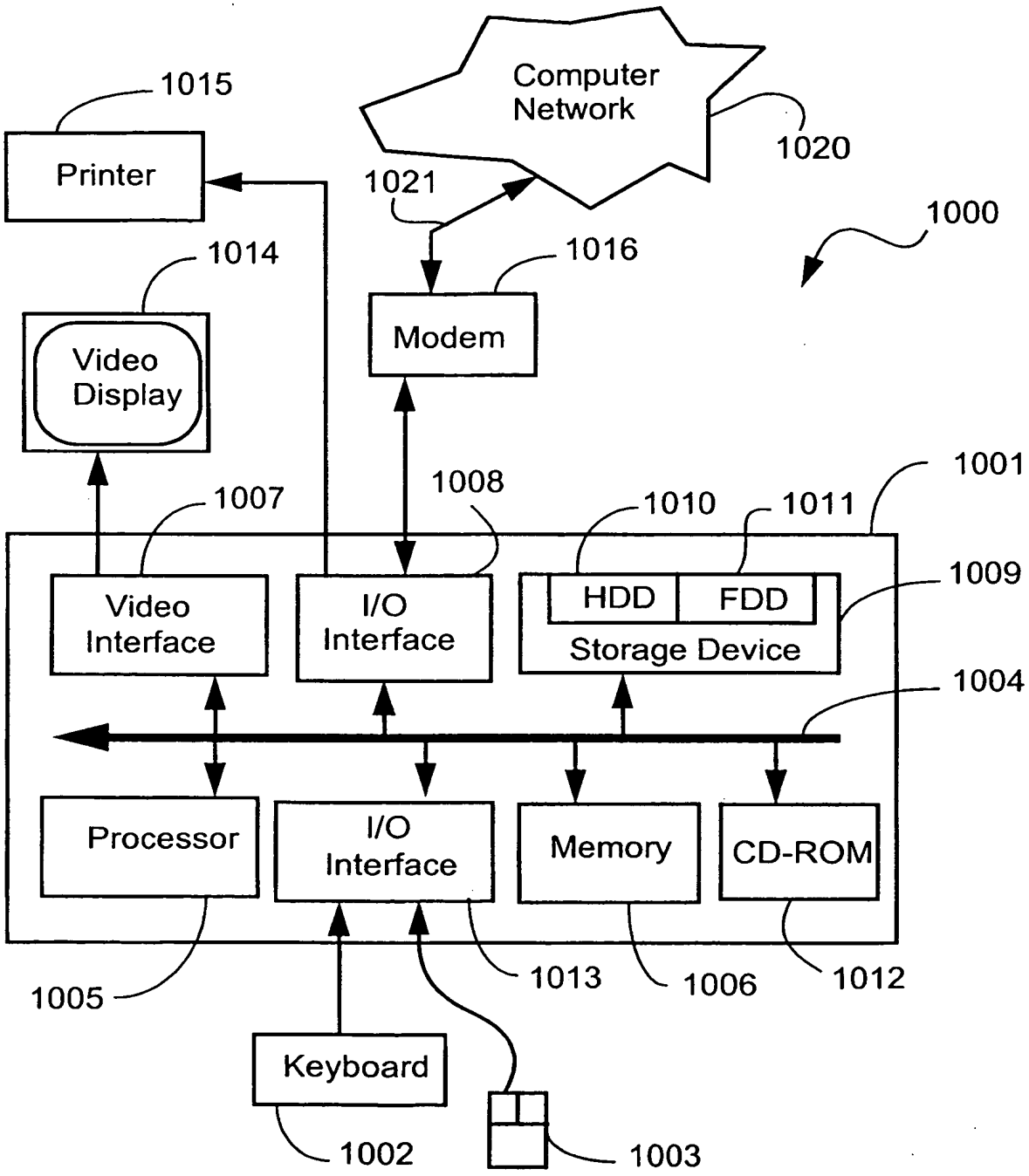
Computer Network
1020
1015
Printer
1021
1000
1014
1016
Video Display
Modem
1001
1008
1007
1010
1011
1009
Video Interface
I/O Interface
HDD
FDD
Storage Device
1004
Processor
I/O Interface
Memory
CD-ROM
1005
1013
1006
1012
Keyboard
1002
1003

Fig. 10